## REMARKS

The Examiner is thanked for the performance of a thorough search.

## STATUS OF CLAIMS

Claims 1-35 have been cancelled.

Claims 36-65 have been added.

No claims have been amended or withdrawn.

Claims 36-65 are currently pending in the application.

## SUMMARY OF THE REJECTIONS/OBJECTIONS

Claims 1-11, 14-23, and 26-33 have been rejected under 35 U.S.C. § 102(b) as allegedly anticipated by International Publication Number WO 00/75783 A1 by Gregory et al. (" *Gregory* "). Claims 12, 13, 24, 25, 34, and 35 have been rejected under 35 U.S.C. § 103(a) as allegedly unpatentable over *Gregory* in view of U.S. Patent Application Publication Number US 2003/0167422 A1 issued to Morrison et al. (" *Morrison* "). Claims 1-35 have been cancelled. Thus, withdrawal of these rejections is respectfully requested.

### A. CLAIM 36

#### (1) INTRODUCTION TO CLAIM 36

Claim 36 features:

"A computer-implemented method for testing an operating system, comprising:

(a) causing said operating system to create **a process** **having at least a first thread and a second thread**;

(b) causing **two or more test functions** to be executed in **said first thread**, wherein each test function of said two or more test functions is a ***different test function***

(c) causing said **two or more test functions** to be executed in **said second thread**; and

(d) repeating steps (a), (b), and (c) to cause said operating system to create **one or more additional processes** each having **at least two threads** and to cause **said two or more test functions** to be executed in each of said at least two threads." (emphasis added).

Thus, Claim 36 features multiple processes in which each process has at least two threads (e.g., "a process having at least a first thread and a second thread" and "one or more additional processes each having at least two threads"). Claim 36 also features two or more test functions in which each test function is different and in which the two or more test functions are executed in each thread of each process.

For example, in the embodiment illustrated in FIG. 4 of the application, three processes 61, 62, 63 are depicted, each having two threads (611, 612), (621, 622), and (661, 662), respectively. FIG. 4 also illustrates a set of "n" test functions, 51-56, that are executed by threads 611, 612, 621, 622, and 661, 662, and each test function 51-56 is a different test function (e.g., Unit Test_1, Unit Test_2...Unit Test_n). (*See also* Application page 8, lines 8-16.) The approach of Claim 36 illustrates a "multi-thread...and multi-process mode" of operation. (Application, page 9, lines 13-24.) Therefore, Claim 36 is fully supported by the specification, and no new matter is introduced.

### (2) INTRODUCTORY DISCUSSION OF *GREGORY*

*Gregory* discloses an approach for protocol acknowledgement between homogenous systems, in which "a protocol acknowledgement software package...uses an operating system-generated event handle as a member field of a protocol for releasing an execution thread which is waiting for an acknowledgment message from the target device." (Title; Abstract.) The "event handle is placed in a header portion of an acknowledgment message packet...[that] is sent back in the acknowledgment message, and...a receiving thread unblocks any send threads of execution which are waiting for the event handle in the acknowledgment message..." (Abstract.) The purpose of the approach of *Gregory* is to eliminate steps such as "adding an element which associates the execution thread with the message ID in a list, cross-referencing the message ID in a list, and determining which execution thread to release from a list entry." (Page 3, lines 2-5.) Thus, the focus of *Gregory* is on how to simplify the use of acknowledgment messages for releasing execution threads.

### (3) THE OFFICE ACTION'S CITATIONS FROM *GREGORY*

The Office Action states that *Gregory* discloses "repetitively invoking an operating system to create a process with a thread and executing a test function repetitively in the thread...Gregory et al. teaches executing threads to test an operating system (page 16, line 5 to

page 17, line 20)." In addition, the Office Action states that Gregory discloses "a second thread being created upon a first thread request...[because Gregory teaches] a main thread spawning other threads (page 17, lines 13-15)."

However, the cited portions of *Gregory* describe three different threads, namely a broadcast thread, an execution thread, and a main execution thread, each having specific but different functions. (Page 17, lines 4-15.) Specifically, the "broadcast thread updates information such as the device IP, connection type, and available com-ports every ten seconds, sending a broadcast message at the same rate to the network." (Page 17, lines5-8.) The "execution thread waits for a connection attempt from an Engine 3. When the execution thread receives the connections, it spawns another thread, a main execution thread that performs the various functions required." (Page 17, lines12-16.) Thus, in *Gregory*, only the main execution thread executes the test suites. The broadcast thread and the execution thread merely perform supporting functions for the main execution thread, and thus neither the broadcast thread nor the execution thread execute the test suites.

In contrast to the disclosure of *Gregory*, Claim 36 features **at least two processes**, ***each having at least two threads***, in which **each thread executes the two or more test functions**. While *Gregory* discloses the use of three different types of threads, only one thread (e.g., the main execution thread) executes the test suites. Therefore, *Gregory* fails to disclose the use of ***multiple processes in which each process has multiple threads each executing the same test functions***, as featured in Claim 36, because the disclosure in *Gregory* of three different types of threads does not relate to the use of multiple processes each having multiple threads that execute multiple test functions, as featured in Claim 36 of the present application.

(4) CONCLUSION OF DISCUSSION OF CLAIM 36 AND *GREGORY*

Because *Gregory* fails to disclose, teach, suggest, or in any way render obvious "causing said operating system to create **a process having at least a first thread and a second thread**," "causing **two or more test functions** to be executed in **said first thread**," and causing "said operating system to create **one or more additional processes** each having **at least two threads** and to cause **said two or more test functions** to be executed in each of

said at least two threads," the Applicant respectfully submits that, for at least the reasons stated above, Claim 36 is allowable over the art of record and is in condition for allowance.

B. CLAIMS 46 AND 56

Claims 46 and 56 contain features that are the same as those described above with respect to Claim 1, and in particular both feature "causing said operating system to create **a process having at least a first thread and a second thread**," "causing **two or more test functions** to be executed in **said first thread**," and causing "said operating system to create **one or more additional processes** each having **at least two threads** and to cause **said two or more test functions** to be executed in each of said at least two threads," as in Claim 36. Therefore, based on at least the reasons stated above with respect to Claim 36, the Applicant respectfully submits that Claims 46 and 56 are allowable over the art of record and are in condition for allowance.

C. CLAIMS 37-45, 47-55, AND 57-65

Claims 37-45, 47-55, and 57-65 are dependent upon Claims 36, 46, and 56, respectively, and thus include each and every feature of the corresponding independent claims. Therefore, it is respectfully submitted that Claims 37-45, 47-55, and 57-65 are allowable for the reasons given above with respect to Claims 36, 46, and 56. In addition, each of Claims 37-45, 47-55, and 57-65 introduces one or more additional limitations that independently render it patentable.

For example, in Claims 38, 48, and 58 the test functions are executed by the threads sequentially, in Claims 39, 49, and 59 each test function is executed substantially simultaneously by each thread, in Claims 40, 50, and 60 each test function "is executed entirely in a particular thread before said each function begins execution in another thread," and in Claims 41, 51, and 61 each test function is executed randomly in each thread.

In the embodiments described in application, tests can be executed sequentially in different threads according to thread priority and time sharing, an identical test can be authorized to be executed entirely only in one thread before being executed in another thread, and the tests invoked by each thread can be chosen randomly from the available tests. (Application, page 8, lines 8-16, page 10, lines 18-24.) Thus, these additional features of Claims 38-41, 48-51, and 58-61 are fully supported by the specification, and no new matter is

added. Furthermore, none of these features are disclosed, taught, suggested, or obvious in light of the cited art.

As additional examples, in Claims 42, 52, and 62, step (d) is repeated (e.g., the repetition of steps (a), (b), and (c) to cause the creation of additional processes with at least two threads each to execute the test functions) "until a total number of processes created by said operating system is at least equal to a predetermined number of processes." Similarly, in Claims 43, 53, and 63, step (d) is executed "such that a total number of threads per process is at least equal to a predetermined number of threads per process," and in Claims 44, 54, and 64, step (d) is executed "such that a total number of times that each test function of said two or more test functions is executed by each thread is at least equal to a predetermined number of times."

For example, in FIG. 5 of the application, a user chooses the parameters to test the operating system at operation 101, such as the number of processes to be created, the number of threads to be created, and the number of times each test is to be executed, which can be specified in parameters to the test framework when launching a test program and therefore selected and stored in advance, such as in a configuration file. (Application, page 8, lines 18-23, and page 10, lines 12-13; page 11, lines 22-26.) As a result, such parameters can be characterized as predetermined. Thus, these additional features of Claims 42-44, 52-54, and 62-64 are fully supported by the specification, and no new matter is added.

The Office Action states that *Gregory* teaches "allowing the user to control the number of cycles and threads of testing by stopping an infinite cycle or configuring the test using data files (page 13, line 20 – page 14, line 21.)" However, *Gregory's* disclosure that an "infinite loop test runs the selected suites 11 until the user manually hits the Stop button" is different than Claims 44, 54, and 64 that feature executing the test functions by each thread until "a predetermined number of times" is reached. In *Gregory*, there is no predetermined number because the loop is infinite, and the test continues to run until manual action is taken by the user, which means that the length of the test is not predetermined. In contrast, the approach of Claims 44, 54, and 64 executes the two or more test functions until a predetermined number of executions occurs, which means that the length of the test is determined at the start of the test, not during the test as in *Gregory*.

## CONCLUSION

The Applicant believes that all issues raised in the Office Action have been addressed and that allowance of the pending claims is appropriate. After entry of the amendments, further examination on the merits is respectfully requested.

If any applicable fee is missing or insufficient, throughout the pendency of this application, the Commissioner is hereby authorized to any applicable fees and to credit any overpayments to our Deposit Account No. 50-1302.

Respectfully submitted,
HICKMAN PALERMO TRUONG & BECKER LLP

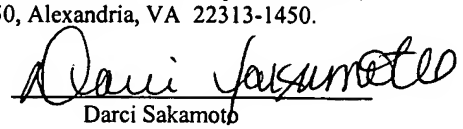
Dated: January 18, 2005

Craig G. Holmes
Reg. No. 44,770

2055 Gateway Place, Suite 550
San Jose, CA 95110-1089
Telephone: (408) 414-1207
Facsimile: (408) 414-1076

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Mail Stop Amendment, Commissioner for Patents, P. O. Box 1450, Alexandria, VA 22313-1450.

on January 18, 2005 by 

Darci Sakamoto